RECEIVED
CENTRAL FAX CENTER
MAR 0 6 2007

REMARKS

Reconsideration of the rejections contained in the Office Action dated November 6, 2006 is respectfully requested. By this amendment, claims 1, 8, 14, and 16-17 have been amended. Currently, claims 1-4, 7-14, and 16-17 are pending in this application.

Objection to claims 1, 2, and 17

The Examiner objected to claims 1, 2, and 17 because, according to the Examiner, several limitations such as "hollowed netlist", "filled netlist", "design constraints", "data-path constraints" and "set of design output files" were unclear and need to be specified clearly in the claim.

The terms objected to by the Examiner are clearly understood by reference to the specification. There is no requirement that the claim terms be defined in the claims. Rather, the claims are sufficiently definite where a person skilled in the art is capable of understanding the claims when read in the context of the specification. In this instance, applicants respectfully submit that the claim terms objected to by the Examiner are clearly explained in the specification.

For example, at paragraph 10 (page 4) applicants explain that initially the functionality to be programmed into an FPGA is described in register transfer language (RTL). The RTL source files are synthesized into a representation of the design made of the interconnection of logic elements. The result of the synthesis process is referred to as a "netlist." Thus, the term netlist is a defined term as set forth in paragraph 4.

The term "hollowed netlist" and "filled netlist" are explained at paragraph 27: "The filled netlist 114 contains the full implementation of the design as described in the original source files 100, but the hollowed netlist 115 only contains the boundaries of the defined logical groups with no internal control logic."

Paragraph 27 also explains that "the design constraints 118 represent the design specifications as originally intended, where the data path constraints 120 only describe specification for signals between logic groups."

The term "data path constraints" is defined at paragraph 26 (page 7) as "top level constraints that describe the timing requirements of signals between logic groups, the characteristics of device pins, and other types of physical limitations."

The "set of design output files" are the scripts, setup files, tool lineup files, etc. that will ultimately be used to program the FPGA. (page 11, paragraph 44).

Applicants note that many of the terms referred to by the Examiner as being unclear in the claim appear to be terms of art within the field of FPGA design. For example, Mason (cited by the Examiner in connection with rejecting the claims) defines the term "netlist" at col. 6, lines 57-61, defines the term constraints at col. 5, lines 45-47, and refers to the output files as the files that contain the configuration data for the programmable logic device (see claim 72). Thus, applicants respectfully submit that the claims are sufficiently specific, given the teachings of the specification and the fact that the terms that were objected to by the Examiner are well understood by those of skill in the art. Accordingly, applicants respectfully request the Examiner to withdraw the objection to claims 1-2 and 17.

Rejection under 35 USC 102

Claims 1-5 and 12-15 were rejected under 35 USC 102 as anticipated by Mason, et al. (U.S. Patent No. 6,817,005). This rejection is respectfully traversed in view of the amendments to the claims and the following arguments.

This application relates to a way of automating the design of a FPGA. Specifically, as claimed in claim 1, the method performs the usual steps of placing the logic blocks, etc. However, applicants discovered that it was possible to iterate one or more of the steps of the FPGA design to have a computer program, for example, design the FPGA with little input from the user.

Mason teaches a standard way of designing a FPGA that requires direct user involvement. If an initial FPGA design does not meet one or more of the timing requirements, the user may adjust one or more parameters, based on their experience, and then cause the new FPGA design to be tested. The process in Mason is not automated the way the disclosed and claimed process is, but rather requires the user to adjust things so that the FPGA design may be created.

Mason teaches a way to enable a portion of an FPGA to be designed independently of the other portions of the FPGA design. Specifically, Morgan states that one problem with FPGA design is that there is a need for logic designers to be able to work in parallel on smaller modules that will be used to build a larger FPGA. (See Col. 1, lines 60-64). With this problem in mind,

Mason goes on to teach a possible way for a larger FPGA design problem to be broken into smaller modules that may be individually designed and tested. The modules may then be combined at a latter time to create the larger FPGA.

The way in which this is done, is that the top level design is first partitioned into smaller blocks, and information generated during the partitioning is used to guide implementation of the associated logic in the top level design of the smaller modules. (See e.g. col. 2, lines 24-28). The constraints that are generated during the partitioning phase are thus used as constraints on the individual modules to enable the modules to be created in parallel.

Mason teaches a three phase design phase, including an initial phase, an active module phase, and an assembly phase. (See Col. 2, lines 54-57). During the initial phase, the top level design is divided into modules, and the size and location of the modules is defined. (Col. 2, lines 58-21). The initial phase also defines the input/output nets for each module. The module is then positioned on the FPGA and module ports are defined. The module ports provide module input/output signals to enable the module to communicate with the other modules. (Col. 3, lines 13-20). Pseudo logic is defined for the ports, which may be placed by the user (constrained pseudologic) or may be placed by a mapper (unconstrained pseudologic).

In the active phase, each module is implemented independently of the other modules. The active phase includes the module build stage, the module map stage, the module place and route stage, the module floorplan stage, the module back annotation state, and the module publishing stage. (Col. 3, lines 55-60). The first three stages are described at col. 3, line 61-col. 4, line 26. However, at col. 4, lines 26-30, Mason states that if the implementation of the module is not correct after completion of these first three stages, the "designer first runs the floorplanner tool to directly place the unconstrained pseudologic and subsequently runs the floorplanner tool to place the module logic." The fact that the designer makes a decision to run the floorplanner tool supports applicants' position that Mason does not teach an automated process of implementing an FPGA design.

Mason describes the assembly phase at Col. 4, lines 44-67. Once again, mason describes the process as requiring the designer to run various tools on the files to generate the module design. More specifically, Mason states at col. 4, lines 61-65, that "If the top-level design is incorrect or fails to meet timing constraints, then the designer can manually designate the appropriate tool for correction. Mapping, placing, and routing are performed again, as

necessary." It is clear from this passage, and from the fact that the various computer programs being used are referred to as "tools" that Mason is describing a manual process in which a designer serially runs various computer programs (tools) to design a module, and then, if an initial design does not meet the various requirements (constraints) the tools are re-run using different parameters specified by the user until the design is successful.

Applicant discovered, by contrast, that it was possible to automate the process of designing a programmable logic device by causing design software to iterate design calculations using slightly different values without substantial intervention from a user until the placement of logic groups meets design constraints. Mason does not teach or suggest an automated process of this nature and specifically does not teach or suggest that the software tools being used to create the FPGA design should automatically try slightly different values until an acceptable design has been achieved. Rather, Mason relies on the designer to adjust the design parameters and then requires the designer to re-run the new adjusted design to see if the adjusted design would meet the timing and other requirements. This was precisely what applicants were trying to avoid (see Specification at page 4 – Paragraph 12). Accordingly, Mason does not teach the solution proposed by applicants but rather teaches a way to use the previously known manual FPGA design techniques to create smaller FPGA modules that may be then later grouped together to form on entire FPGA.

Applicants have amended the claims to more clearly recite the manner in which the iteration step occurs. Specifically, applicants have amended claim 1 to recite that the process iterates "using adjusted values without substantial intervention from a user" until the placement of the logic groups meets the design constraints. Since Mason does not teach or suggest a process of this nature, applicants respectfully request that the rejection of claim 1 be withdrawn. Similar amendments have been made to the other independent claims rendering them patentable for the same reason.

Conclusion

In view of foregoing claim amendments and remarks, it is respectfully submitted that the application is now in condition for allowance and an action to this effect is respectfully requested. If there are any questions or concerns regarding the amendments or these remarks, the Examiner is requested to telephone the undersigned at the telephone number listed below.

Amendment Date: March 6, 2007
Serial No. 10/771,596

If any fees are due in connection with this filing, the Commissioner is hereby authorized to charge payment of the fees associated with this communication or credit any overpayment to Deposit Account No. 502246 (Ref: NN-16550).

Respectfully Submitted

Dated: March 6, 2007

John C. Gorecki
Registration No. 38,471

John C. Gorecki, Esq.
P.O. Box 553
Carlisle, MA 01741
Tel: (978) 371-3218
Fax: (978) 371-3219

john@gorecki.us